

Linear Combinations of Unordered Data Vectors

Piotr Hofman^{*1}, Jérôme Leroux², and Patrick Totzke^{†3}

1 LSV, CNRS & ENS Cachan, France

2 LaBRI, CNRS, France

3 LFCS, University of Edinburgh, UK

Abstract

Data vectors generalise finite multisets: they are finitely supported functions into a commutative monoid. We study the question if a given data vector can be expressed as a finite sum of others, only assuming that 1) the domain is countable and 2) the given set of base vectors is finite up to permutations of the domain.

Based on a succinct representation of the involved permutations as integer linear constraints, we derive that positive instances can be witnessed in a bounded subset of the domain.

For data vectors over a group we moreover study when a data vector is reversible, that is, if its inverse is expressible using only nonnegative coefficients. We show that if all base vectors are reversible then the expressibility problem reduces to checking membership in finitely generated subgroups. Moreover, checking reversibility also reduces to such membership tests.

These questions naturally appear in the analysis of counter machines extended with unordered data: namely, for data vectors over $(\mathbb{Z}^d, +)$ expressibility directly corresponds to checking state equations for Coloured Petri nets where tokens can only be tested for equality. We derive that in this case, expressibility is in NP, and in P for reversible instances. These upper bounds are tight: they match the lower bounds for standard integer vectors (over singleton domains).

1998 ACM Subject Classification F.1.1 Models of Computation

Keywords and phrases Computation with atoms, vector addition systems

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Finite collections of named values are basic structures used in many areas of theoretical computer science. They can be used for instance to model databases snapshots or define the operational semantics of programming languages. We can formalize these as functions $\mathbf{v} : \mathbb{D} \rightarrow X$ from some countable domain \mathbb{D} of *names* or *data*, into some value space X , and call such functions (X -valued) *data vectors*. Often the actual names used are not relevant and instead one is interested in data vectors *up to renaming*, i.e., one wants to consider vectors \mathbf{v} and \mathbf{w} equivalent if $\mathbf{v} = \mathbf{w} \circ \theta$ for some permutation $\theta : \mathbb{D} \rightarrow \mathbb{D}$ of the domain.

We consider the case where the value space X has additional algebraic structure. Namely, we focus on data vectors where the values are from some commutative monoid $(M, +, 0)$ and where all but finitely many names are mapped to the neutral element. A natural question then asks if a given data vector is expressible as a sum of vectors from a given set, where the monoid operation is lifted to data vectors pointwise.

* Supported by Labex Digicosme, Univ. Paris-Saclay, project VERICONISS and by Polish NSC grant 2013/09/B/ST6/01575.

† Supported by the EPSRC, grant EP/M027651/1.



© Piotr Hofman, Jérôme Leroux and Patrick Totzke;
licensed under Creative Commons License CC-BY

Conference title on which this volume is based on.

Editors: Billy Editor and Bill Editors; pp. 1–17



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

If the spanning set is finite only up to permutations, this problem does not immediately boil down to solving finite system of linear equations. Also, one cannot simply lift operations on data vectors to equivalence classes of data vectors because the result of pointwise applying the operation depends on the chosen representants. For example, if we have data vectors mapping colours to integers, then the vector $\begin{smallmatrix} \text{blue} \\ \text{red} \end{smallmatrix}$ ($\text{red} \mapsto -1, \text{blue} \mapsto 1$) is equivalent to $\begin{smallmatrix} \text{blue} \\ \text{yellow} \end{smallmatrix}$. Yet still,

$$\begin{smallmatrix} \text{red} & \text{yellow} \\ \text{blue} & \end{smallmatrix} + \begin{smallmatrix} \text{blue} \\ \text{red} \end{smallmatrix} = \begin{smallmatrix} \text{yellow} \\ \end{smallmatrix} \neq \begin{smallmatrix} \text{red} & \text{yellow} \\ \end{smallmatrix} = \begin{smallmatrix} \text{red} & \text{yellow} \\ \text{blue} & \end{smallmatrix} + \begin{smallmatrix} \text{blue} \\ \text{yellow} \end{smallmatrix}.$$

We thus choose to keep permutations explicit and consider the *Expressibility* problem:

Input: A finite set V of data vectors and a target vector \mathbf{x} .

Question: does \mathbf{x} equal $\sum_{i=1}^k \mathbf{v}_i \circ \theta_i$ for some $\mathbf{v}_i \in V$ and permutations $\theta_i : \mathbb{D} \rightarrow \mathbb{D}$?

If the domain \mathbb{D} is finite then this just asks if some finite system of linear equations is satisfiable. Over infinite domains this corresponds to find a solution of an infinite but regular set of linear equations. For brevity, we will call a vector \mathbf{x} a *permutation sum* of V if it is expressible as sum of permutations of vectors in V as above.

Contributions and Outline. We provide two reductions from the Expressibility problem to problems of finding solutions for *finite* linear systems over $(M, +)$.

The more general approach is presented in Sections 3 and 4 and ultimately works by bounding the number of different data values necessary to express a permutation sum. This is based on an analysis of objects we call *histograms*, see Section 3, which sufficiently characterize vectors expressible as permutation sums of single vectors. For any monoid $(M, +)$ the Expressibility problem then reduces to finding a non-negative integer solution of a finite system of linear inequations over $(M, +)$. In particular, for monoids $(\mathbb{Z}^d, +)$ this provides an NP algorithm, matching the lower bound from the feasibility of integer linear programs.

The second approach (Section 5) reduces Expressibility to the problem of finding an (not necessarily non-negative) integer solution to a finite linear system. This assumes that $(M, +)$ is a group and that all base vectors are reversible, i.e., their inverses are expressible. We show that this reversibility condition can be verified by checking the existence of rational solutions of a system over $(M, +)$. For monoids $(\mathbb{Z}^d, +)$, checking this reversibility condition and solving the Expressibility problem for reversible instances is possible in deterministic polynomial time.

We show two applications to the reachability analysis of counter programs extended with data (in Section 6). The first involves finding state invariants for unordered data Petri nets [22, 33, 18] and the second application is the reachability problem for blind counter automata [15] extended with data.

Related Research and Motivation. Our main motivation for studying the Expressibility problem comes from the analysis of Petri nets extended with data – the model of *Unordered Petri data nets* (UPDN) of [22], discussed in Section 6.1 – where data vectors of the form $\mathbf{v} : \mathbb{D} \rightarrow \mathbb{Z}^d$ occur naturally. We are interested in an invariant sometimes called *state equations* in the Petri net literature.

State equation [27] is a fundamental invariant of the reachability relation for Petri nets and one of the important ingredients in the proof of decidability of the reachability relation [26, 21, 24]. Some modification of it can be also used as heuristic to improve a performance of the standard backward coverability algorithm for Petri nets (due to well-structured transition

systems [14, 1, 35, 2]) like it was done in [6]. The coverability problem has been proven to be EXPSPACE-complete in [28, 25] and in [9] it was shown that the backward algorithm matches this complexity.

UDPN were introduced in [22] as one of the extensions of Petri nets in which the coverability problem remains decidable. Due to results about undecidability of boundedness for Petri nets with resetting arcs [12], it is not hard to conclude that UDPN is the only extension of Petri nets, among those proposed in [22], for which reachability problem may be decidable. The first indicator that reachability may be decidable for UDPN is a characterization of the coverability set established in the paper [18], in the same paper, as a conclusion, the place-boundedness problem is proven to be decidable. The recent development in other classes proposed in [22] can be found in papers [33, 32, 31, 18], all those results are focused on better understanding of the coverability relation.

UDPN can also be seen as a restriction of more general *Colored Petri nets* (CPN), see for instance [19]. There is a long history of research in the area of restricted CPNs. Here, we point to results about invariants and identification of certain syntactic substructures: in [17, 13] authors investigate flows in subclasses CPN. Another important branch of research concerns structural properties of Algebraic Nets [30] like detecting siphons [34] or other kind of place invariants [37].

The third perspective is algebraic methods for Petri nets. Linear algebra and linear programming are one of the most fruitful approaches to Petri nets. A broad overview of algebraic methods for Petri nets can be found in [36]. A beautiful application of algebraic techniques are results on reachability in continuous Petri nets [29]. One can also find variants of state equation for Petri nets with resetting transitions [16] or with inhibitor arcs [4]. Finally, algebraic methods are also used in restricted classes of Petri nets like conflict-free and free-choice Petri nets [10].

Finally, it is worth mentioning that UDPN can be interpreted as ordinary Petri nets for sets with *equality atoms*. We refer the reader to [8, 7] for work on sets with atoms/nominal sets. Very similar in spirit to our study of data vectors is the work in [20] that considers constraint satisfaction problems on infinite structures.

2 Data Vectors

In the sequel \mathbb{D} is a countable set of elements called *data values* and $(M, +)$ is a commutative monoid with neutral element 0. A *data vector* (also *vector* for short) is a total function $\mathbf{v} : \mathbb{D} \rightarrow M$ such that the *support*, the set $\text{supp}(\mathbf{v}) \stackrel{\text{def}}{=} \{\alpha \in \mathbb{D} \mid \mathbf{v}(\alpha) \neq 0\}$ is finite. The monoid operation $+$ is lifted to vectors pointwise, so that $(\mathbf{v} + \mathbf{w})(\alpha) \stackrel{\text{def}}{=} \mathbf{v}(\alpha) + \mathbf{w}(\alpha)$.

Writing \circ for function composition, we see that $\mathbf{v} \circ \pi$ is a data vector for any data vector \mathbf{v} and permutation $\pi : \mathbb{D} \rightarrow \mathbb{D}$. A vector \mathbf{x} is said to be a *permutation sum* of a set V of vectors if there are $\mathbf{v}_1, \dots, \mathbf{v}_n$ in V and permutations $\theta_1, \dots, \theta_n$ of \mathbb{D} such that

$$\mathbf{x} = \sum_{i=1}^n \mathbf{v}_i \circ \theta_i.$$

Here, we have to emphasize that it is possible that $\mathbf{v}_i = \mathbf{v}_j$ for some i and j .

When working with vectors of the form $\mathbf{v} \circ \theta$ for permutations θ , it will be instrumental to specify θ indirectly using some injection of $\text{supp}(\mathbf{v})$ into \mathbb{D} . In the remainder of this section we show that one can always do this.

Take any finite subset \mathbb{S} of \mathbb{D} and $\pi : \mathbb{S} \rightarrow \mathbb{D}$ injective. Since π^{-1} is only a partial function, define the data vector $\mathbf{v} \circ \pi^{-1}$ so that any β not in the range of π maps to 0: More

precisely, let $(\mathbf{v} \circ \pi^{-1}) : \mathbb{D} \rightarrow M$ be defined, for every $\beta \in \mathbb{D}$ as follows.

$$\mathbf{v} \circ \pi^{-1}(\beta) = \begin{cases} \mathbf{v}(\alpha) & \text{if } \pi(\alpha) = \beta \\ 0 & \text{if } \pi(\alpha) \neq \beta \text{ for all } \alpha \in \mathbb{D} \end{cases}$$

► **Lemma 1.** *Let \mathbf{v} be a data vector. For any permutation $\theta : \mathbb{D} \rightarrow \mathbb{D}$ there exists an injection $\pi : \text{supp}(\mathbf{v}) \rightarrow \mathbb{D}$, such that $\mathbf{v} \circ \pi^{-1} = \mathbf{v} \circ \theta$, and vice versa.*

Proof. Let us introduce $\mathbb{S} = \text{supp}(\mathbf{v})$ and first consider a permutation θ . We show that the injection $\pi : \mathbb{S} \rightarrow \mathbb{D}$, defined by $\pi(\alpha) = \theta^{-1}(\alpha)$ for every $\alpha \in \mathbb{S}$, satisfies $\mathbf{v} \circ \theta = \mathbf{v} \circ \pi^{-1}$.

Pick any $\beta \in \mathbb{D}$ and let us write $\alpha \stackrel{\text{def}}{=} \theta(\beta)$. If $\alpha \in \mathbb{S}$ then $\pi(\alpha) = \beta$ so, we have that $(\mathbf{v} \circ \pi^{-1})(\beta) = \mathbf{v}(\alpha) = (\mathbf{v} \circ \theta)(\beta)$. If $\alpha \notin \mathbb{S}$ then $(\mathbf{v} \circ \pi^{-1})(\beta) = 0$ and $\mathbf{v} \circ \theta(\beta) = \mathbf{v}(\alpha) = 0$, by definition of $\mathbf{v} \circ \pi^{-1}$ and because \mathbb{S} is the support of \mathbf{v} .

Conversely, let us consider a data injection π over \mathbb{S} and let us prove that there exists a data permutation θ such that $\mathbf{v} \circ \pi^{-1} = \mathbf{v} \circ \theta$. We introduce $\mathbb{T} = \pi(\mathbb{S})$. Since the restriction of π on \mathbb{S} is a bijection onto \mathbb{T} , there exists a bijection $\pi' : \mathbb{T} \rightarrow \mathbb{S}$ denoting its inverse. We introduce the sets $X = \mathbb{S} \setminus \mathbb{T}$, and $Y = \mathbb{T} \setminus \mathbb{S}$. Since \mathbb{T} and \mathbb{S} have the same cardinal, it follows that X and Y are two finite sets with the same cardinal. Hence, there exists a bijection $\pi'_{Y,X} : X \rightarrow Y$ and its inverse $\pi_{Y,X} : Y \rightarrow X$. We introduce the function θ defined for every $\beta \in \mathbb{D}$ as follows:

$$\theta(\beta) = \begin{cases} \pi'(\beta) & \text{if } \beta \in \mathbb{T} \\ \pi'_{Y,X}(\beta) & \text{if } \beta \in X \\ \beta & \text{otherwise} \end{cases}$$

Observe that θ is a bijection since the function θ' defined for every $\alpha \in \mathbb{D}$ as follows is its inverse:

$$\theta'(\alpha) = \begin{cases} \pi(\alpha) & \text{if } \alpha \in \mathbb{S} \\ \pi_{Y,X}(\alpha) & \text{if } \alpha \in Y \\ \alpha & \text{otherwise.} \end{cases}$$

We show that $\mathbf{v} \circ \pi^{-1} = \mathbf{v} \circ \theta$. Fix some $\beta \in \mathbb{D}$ and assume first that $\beta \in \mathbb{T}$. There exists $\alpha \in \mathbb{S}$ such that $\pi(\alpha) = \beta$. It follows that $\pi'(\beta) = \alpha = \theta(\beta)$. Hence, $(\mathbf{v} \circ \theta)(\beta) = \mathbf{v}(\alpha) = (\mathbf{v} \circ \pi^{-1})(\beta)$.

Now, assume that $\beta \notin \mathbb{T}$. In that case, notice that $\theta(\beta)$ is not in \mathbb{S} no matter if $\beta \in X$ or not. Thus $\mathbf{v} \circ \theta(\beta) = 0$. Observe that if $\pi^{-1}(\{\beta\})$ is empty, we deduce that $(\mathbf{v} \circ \pi^{-1})(\beta) = 0$. If $\pi^{-1}(\beta) = \{\alpha\}$ then $(\mathbf{v} \circ \pi^{-1})(\beta) = \mathbf{v}(\alpha)$. But since $\beta \notin \mathbb{T}$, we deduce that $\alpha \notin \mathbb{S}$. Therefore $\mathbf{v}(\alpha) = 0$ and we derive that $(\mathbf{v} \circ \pi^{-1})(\beta) = 0$. We have proved that $\mathbf{v} \circ \pi^{-1} = \mathbf{v} \circ \theta$. ◀

3 Histograms

In this section we develop the notion of histograms. These are combinatorial objects that will be used in the next section to characterize permutation sums over singleton sets V .

► **Definition 2.** A *histogram* over a finite set $\mathbb{S} \subseteq \mathbb{D}$ is a total function $H : \mathbb{S} \times \mathbb{D} \rightarrow \mathbb{N}$ such that for some $n \in \mathbb{N}$, called the *degree* of H , the following two conditions hold.

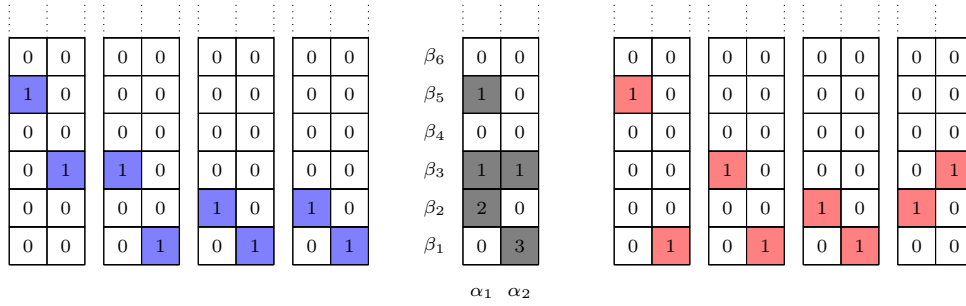
1. $\sum_{\beta \in \mathbb{D}} H(\alpha, \beta) = n$ for any $\alpha \in \mathbb{S}$
2. $\sum_{\alpha \in \mathbb{S}} H(\alpha, \beta) \leq n$ for any $\beta \in \mathbb{D}$.

A histogram of degree $n = 1$ is called *simple*. Histograms with the same signature, i.e. the same sets \mathbb{S} and \mathbb{D} , can be partially ordered and summed pointwise and the degree of the sum is the sum of degrees. The *support* of H is the set $\text{supp}(H) \stackrel{\text{def}}{=} \{\beta \mid \sum_{\alpha \in \mathbb{S}} H(\alpha, \beta) > 0\}$.

The following theorem states the main combinatorial property we are interested in, namely that simple histograms over \mathbb{S} generate as finite sums the class of all histograms over \mathbb{S} . In particular, any histogram can be decomposed into finitely many simple histograms over the same signature (see Figure 1 for an illustration).

► **Theorem 3.** *A function $H : \mathbb{S} \times \mathbb{D} \rightarrow \mathbb{N}$ is a histogram of degree $n \in \mathbb{N}$ if, and only if, H is the sum of n simple histograms over \mathbb{S} .*

■ **Figure 1** The center depicts a histogram $H : \mathbb{S} \times \mathbb{D} \rightarrow \mathbb{N}$ of degree 4, where $\mathbb{S} = \{\alpha_1, \alpha_2\}$ and $\text{supp}(H) = \{\beta_1, \beta_2, \beta_3, \beta_5\}$. To the left (in blue) and to the right (in red) are decompositions into four simple histograms each.



For the proof of Theorem 3 we need a lemma from graph theory. We refer the reader to [11] for relevant definitions and recall here only that in a graph (V, E) , a *matching* of a set $S \subseteq V$ of nodes is a set $M \subseteq E$ of pairwise non-adjacent edges that covers all nodes in S .

► **Lemma 4.** *Let $G = (L \cup R, E)$ be a bipartite graph. If there is a matching of $L' \subseteq L$ and a matching of $R' \subseteq R$ then there is a matching of $L' \cup R'$.*

Proof. Suppose, M_L and M_R are matchings that matches L' and R' , respectively. Let $G' \stackrel{\text{def}}{=} (L \cup R, M_L \cup M_R)$ be a subgraph of G . We construct a matching M of $L' \cup R'$ as a matching in G' . Observe that G' is a union of single nodes, paths and cycles and M can be constructed in every strongly connected component independently.

We claim that for any strongly connected component C we can find a matching witch matches all elements in $C \cap L'$ and $C \cap R'$. This, if proved, ends the proof of Lemma 4.

First of all, every node in $L' \cup R'$ has a degree at least 1 so the claim holds for single nodes immediately.

If the strongly connected component is a cycle (in bipartite graph) or a path of an even length then there is a perfect matching in it so the claim holds as well.

The case of paths of odd length is the most complicated one. Without loosing of generality, suppose that the first node x is in $L' \cup R'$. Indeed, if it is not then we match all nodes except x and this case is done.

Without loss of generality, we can assume that $x \in L'$, as $x \in R'$ is symmetric. We prove that the path has to end in vertex from the set $L \setminus L'$. Indeed path has to end in L as it's length is odd. Furthermore, consider a walk along the path starting from x ; to every element of $L' \setminus \{x\}$ on the path C we enter via an edge from M_R , but then we can leave it via an edge from M_L as from any vertex in L' there is outgoing edge in M_L . Thus, the path

can not end in the element from L' and thus the last vertex has to belong to $L \setminus L'$. Now we match all vertices except of the last one. \blacktriangleleft

Proof of Theorem 3. If $H = \sum_{j=1}^n H_j$, where the H_j are simple histograms over \mathbb{S} , then from $\sum_{\beta \in \mathbb{D}} H(\alpha, \beta) = \sum_{j=1}^n \sum_{\beta \in \mathbb{D}} H_j(\alpha, \beta)$ and because $\sum_{\beta \in \mathbb{D}} H_j(\alpha, \beta) = 1$ we derive that $\sum_{\beta \in \mathbb{D}} H(\alpha, \beta) = n$ for every $\alpha \in \mathbb{D}$. In the same way the second histogram condition for H follows from those of the H_j . So H is a histogram over \mathbb{S} .

For the converse direction, the proof proceeds by induction on n , the degree of the histogram H . If $n = 1$ then H is simple and the claim trivially holds. Suppose now that the claim holds for histograms of degree n and consider a histogram $H : \mathbb{S} \times \mathbb{D} \rightarrow \mathbb{N}$ of degree $n + 1$. We show how that there exists a simple histogram $X : \mathbb{S} \times \mathbb{D} \rightarrow \mathbb{N}$ such that

1. $X(\alpha, \beta) \leq H(\alpha, \beta)$ for every $\alpha \in \mathbb{S}$, and every $\beta \in \mathbb{D}$, and
2. for every $\beta \in \mathbb{D}$ if $\sum_{\alpha \in \mathbb{S}} H(\alpha, \beta) = n + 1$ then $\sum_{\alpha \in \mathbb{S}} X(\alpha, \beta) = 1$.

The first condition then guarantees that the function $Y : \mathbb{S} \times \mathbb{D} \rightarrow \mathbb{N}$ with $Y = H - X$ is well defined and satisfies

$$\sum_{\beta \in \mathbb{D}} Y(\alpha, \beta) = \left(\sum_{\beta \in \mathbb{D}} H(\alpha, \beta) \right) - \left(\sum_{\beta \in \mathbb{D}} X(\alpha, \beta) \right) = (n + 1) - 1 = n \quad \text{for all } \alpha \in \mathbb{S}.$$

The second condition implies that $\sum_{\alpha \in \mathbb{S}} Y(\alpha, \beta) \leq n$ for all $\beta \in \mathbb{D}$. Hence, Y is a histogram of degree n and the claim follows by induction hypothesis.

To show the existence of a suitable simple histogram X , we consider now the bipartite graph G where the sets of nodes are \mathbb{S} and $\mathbb{B} \stackrel{\text{def}}{=} \text{supp}(H)$ and where there is an edge between α and β whenever $H(\alpha, \beta) > 0$ (We assume here w.l.o.g. that \mathbb{S} and $\text{supp}(H)$ are disjoint; otherwise take \mathbb{B} as some suitable duplication). Moreover, let \mathbb{T} denote the set of those “maximal” data values β where $\sum_{\alpha \in \mathbb{D}} H(\alpha, \beta) = n + 1$. Note that $\mathbb{T} \subseteq \mathbb{B}$.

We claim that the required simple histogram X exists iff there is a matching in the graph G that matches both \mathbb{S} and \mathbb{T} . Indeed, any such histogram X provides a matching $M \stackrel{\text{def}}{=} \{(\alpha, \beta) \mid X(\alpha, \beta) = 1\}$. By the first histogram condition, M matches all nodes in \mathbb{S} ; and all nodes $\beta \in \mathbb{T}$ are matched since X must satisfy $\sum_{\alpha \in \mathbb{S}} X(\alpha, \beta) = 1$. Conversely, for a given matching M we define $X(\alpha, \beta) = 1$ if $(\alpha, \beta) \in M$ and 0 otherwise. It is easy to check that X is a simple histogram that satisfies required properties.

To finish the proof we show that a matching M of $\mathbb{S} \cup \mathbb{T}$ exists. By Lemma 4, it suffices to find two matchings, one of \mathbb{S} and one of \mathbb{T} . In both cases, we will make use of Hall’s marriage Theorem [11]. Writing $Nb(\mathbb{S})$ for the *neighbourhood* of a set \mathbb{S} of nodes (the set of nodes $v \notin \mathbb{S}$ adjacent to some node in \mathbb{S}), this theorem states that in a finite bipartite graph there is a matching of a set \mathbb{S} of nodes if, and only if every subset $\mathbb{S}' \subseteq \mathbb{S}$ has at least as many neighbours as elements:

$$|\mathbb{S}'| \leq |Nb(\mathbb{S}')|. \tag{1}$$

We start by proving the existence of a matching of \mathbb{S} . If we label the edges (α, β) in our graph by the respective values $H(\alpha, \beta)$, then we observe that the total weight of edges connecting any subset $\mathbb{S}' \subseteq \mathbb{S}$ is at most the total weight of edges connecting its neighbours: $\sum_{\alpha \in \mathbb{S}'} \sum_{\beta \in \mathbb{D}} H(\alpha, \beta) \leq \sum_{\alpha \in \mathbb{S}'} \sum_{\beta \in Nb(\mathbb{S}')} H(\alpha, \beta)$. Consequently,

$$|\mathbb{S}'| \cdot (n+1) = \sum_{\substack{\alpha \in \mathbb{S}' \\ \beta \in \mathbb{D}}} H(\alpha, \beta) \leq \sum_{\substack{\alpha \in \mathbb{S} \\ \beta \in Nb(\mathbb{S}')}} H(\alpha, \beta) \leq \sum_{\beta \in Nb(\mathbb{S}')} (n+1) = |Nb(\mathbb{S}')| \cdot (n+1).$$

The first equality is due to the first histogram condition and the second inequality is by the second histogram condition. So all subsets $\mathbb{S}' \subseteq \mathbb{S}$ satisfy Equation (1), so Hall's theorem applies and there exists a matching of \mathbb{S} .

The proof that a matching of \mathbb{T} exists follows the same pattern. For any subset $\mathbb{T}' \subseteq \mathbb{T}$ we get

$$|\mathbb{T}'| \cdot (n+1) = \sum_{\beta \in \mathbb{T}'} \sum_{\alpha \in \mathbb{S}} H(\alpha, \beta) \leq \sum_{\alpha \in Nb(\mathbb{T}')} \sum_{\beta \in \mathbb{D}} H(\alpha, \beta) = |Nb(\mathbb{T}')| \cdot (n+1),$$

where the first equality holds by the definition of \mathbb{T} . So \mathbb{T} satisfies the assumption of Hall's theorem and we conclude that some matching of \mathbb{T} exists, as required. \blacktriangleleft

4 Expressibility

In this section, we show that histograms provide a natural tool for deciding if a data vector is a permutation sum of others. We first establish the connection between histograms and permutation sums, and then (in Theorem 7) that permutation sums can be represented by histograms with bounded support sets. Finally, we derive an NP complexity upper bound for the Expressibility problem for vectors over monoids $(\mathbb{Z}^d, +)$.

Given a data vector \mathbf{v} and a histogram H over $\mathbb{S} \stackrel{\text{def}}{=} \text{supp}(\mathbf{v})$ we define the vector $\text{eval}(\mathbf{v}, H)$ by

$$\text{eval}(\mathbf{v}, H)(\beta) \stackrel{\text{def}}{=} \sum_{\alpha \in \mathbb{S}} \mathbf{v}(\alpha) H(\alpha, \beta).$$

Observe that eval is a homomorphism in the sense that for any vector \mathbf{v} and histograms H_1, H_2 over \mathbb{S} it holds that

$$\text{eval}(\mathbf{v}, H_1 + H_2) = \text{eval}(\mathbf{v}, H_1) + \text{eval}(\mathbf{v}, H_2). \quad (2)$$

Recall that by Lemma 1, permutation sums are of the form $\mathbf{x} = \sum_{i=1}^n \mathbf{v}_i \circ \pi_i^{-1}$ where $\pi_i : \text{supp}(\mathbf{v}_i) \rightarrow \mathbb{D}$ are injections. We now associate each injective function $\pi : \mathbb{S} \rightarrow \mathbb{D}$ with the simple histogram H_π over \mathbb{S} defined as

$$H_\pi(\alpha, \beta) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } \beta = \pi(\alpha) \\ 0 & \text{otherwise} \end{cases}$$

Notice that conversely, each simple histogram $H : \mathbb{S} \times \mathbb{D} \rightarrow \mathbb{D}$ provides a unique injection $\pi_H : \mathbb{S} \rightarrow \mathbb{D}$ satisfying $H(\alpha, \pi_H(\alpha)) = 1$ for every $\alpha \in \mathbb{S}$. So, $H_{\pi_H} = H$. The next lemma makes the connection between histograms and permutation sums.

► **Lemma 5.** *Let \mathbf{v} be a vector and $\pi : \text{supp}(\mathbf{v}) \rightarrow \mathbb{D}$ injective. Then $\mathbf{v} \circ \pi^{-1} = \text{eval}(\mathbf{v}, H_\pi)$.*

Proof. If $\pi(\alpha) = \beta$ then $\mathbf{v} \circ \pi^{-1}(\beta) = \mathbf{v}(\alpha) = \sum_{\alpha' \in \mathbb{S}} \mathbf{v}(\alpha') H(\alpha', \beta) = \text{eval}(\mathbf{v}, H)(\beta)$ where the second equation holds because H is simple. If $\pi(\alpha) \neq \beta$ for all α then $\mathbf{v} \circ \pi^{-1}(\beta) = 0 = \sum_{\alpha \in \mathbb{S}} \mathbf{v}(\alpha) \cdot 0 = \text{eval}(\mathbf{v}, H)(\beta)$. \blacktriangleleft

► **Lemma 6.** *Let \mathbf{v} be a vector. A vector \mathbf{x} is a permutation sum of $\{\mathbf{v}\}$ if, and only if, there exists a histogram H over $\text{supp}(\mathbf{v})$ such that $\mathbf{x} = \text{eval}(\mathbf{v}, H)$.*

Proof. Assume first that \mathbf{x} is a permutation sum of $\{\mathbf{v}\}$. Then there are permutations $\theta_1, \dots, \theta_n$ such that $\mathbf{x} = \sum_{j=1}^n \mathbf{v} \circ \theta_j$. By Lemma 1, $\mathbf{x} = \sum_{j=1}^n \mathbf{v} \circ \pi_j^{-1}$ for injections $\pi_j : \text{supp}(\mathbf{v}) \rightarrow \mathbb{D}$. From Lemma 5 we derive $\mathbf{x} = \sum_{j=1}^n \text{eval}(\mathbf{v}, H_{\pi_j})$ and by Equation (2) the histogram $H \stackrel{\text{def}}{=} \sum_{j=1}^n H_{\pi_j}$ satisfies $\mathbf{x} = \text{eval}(\mathbf{v}, H)$.

Conversely, let us assume that there exists a histogram H over $\text{supp}(\mathbf{v})$ such that $\mathbf{x} = \text{eval}(\mathbf{v}, H)$. Theorem 3 shows that H can be decomposed as $H = \sum_{j=1}^n H_j$ where H_j are simple histograms over $\text{supp}(\mathbf{v})$. From Equation (2) it follows that $\mathbf{x} = \sum_{j=1}^n \text{eval}(\mathbf{v}, H_j)$, and since all H_j are simple, there are injections π_1, \dots, π_n with $H_j = H_{\pi_j}$. The claim thus follows by Lemma 5 and Lemma 1. \blacktriangleleft

We now show how to bound the supports of histograms $H_{\mathbf{v}}$ such that $\mathbf{x} = \sum_{\mathbf{v} \in V} \text{eval}(\mathbf{v}, H_{\mathbf{v}})$ with respect to \mathbf{x} and V .

► **Theorem 7.** *If \mathbf{x} is a permutation sum of V then $\mathbf{x} = \sum_{\mathbf{v} \in V} \text{eval}(\mathbf{v}, H_{\mathbf{v}})$ where for each vector $\mathbf{v} \in V$, $H_{\mathbf{v}}$ is a histogram over $\text{supp}(\mathbf{v})$, and $|\bigcup_{\mathbf{v} \in V} \text{supp}(H_{\mathbf{v}})|$ is bounded by $|\text{supp}(\mathbf{x})| + 1 + \sum_{\mathbf{v} \in V} (2|\text{supp}(\mathbf{v})| - 1)$.*

Proof. Any permutation sum of V is a sum $\mathbf{x} = \sum_{\mathbf{v} \in V} \mathbf{x}_{\mathbf{v}}$, where each $\mathbf{x}_{\mathbf{v}}$ is a permutation sum of $\{\mathbf{v}\}$. So the existence of histograms $H_{\mathbf{v}}$ with $\mathbf{x}_{\mathbf{v}} = \text{eval}(\mathbf{v}, H_{\mathbf{v}})$ is guaranteed by Lemma 6.

For each histogram $H_{\mathbf{v}}$ we write $n_{\mathbf{v}}$ for its degree and define the set of *big* data values as

$$\mathbb{B}_{\mathbf{v}} \stackrel{\text{def}}{=} \left\{ \beta \in \mathbb{D} \mid \sum_{\alpha \in \text{supp}(\mathbf{v})} H_{\mathbf{v}}(\alpha, \beta) > \frac{n_{\mathbf{v}}}{2} \right\}.$$

We can estimate the cardinality of $\mathbb{B}_{\mathbf{v}}$ by $|\mathbb{B}_{\mathbf{v}}| \leq 2|\text{supp}(\mathbf{v})| - 1$. Indeed, if $\mathbb{B}_{\mathbf{v}}$ is empty, the property is immediate. Otherwise, we have $\sum_{\beta \in \mathbb{B}_{\mathbf{v}}} \sum_{\alpha \in \text{supp}(\mathbf{v})} H_{\mathbf{v}}(\alpha, \beta) > |\mathbb{B}_{\mathbf{v}}| \frac{n_{\mathbf{v}}}{2}$. We also have $\sum_{\beta \in \mathbb{B}_{\mathbf{v}}} \sum_{\alpha \in \text{supp}(\mathbf{v})} H_{\mathbf{v}}(\alpha, \beta) \leq \sum_{\alpha \in \text{supp}(\mathbf{v})} \sum_{\beta \in \mathbb{D}} H_{\mathbf{v}}(\alpha, \beta) = |\text{supp}(\mathbf{v})| \cdot n_{\mathbf{v}}$. Therefore, $|\mathbb{B}_{\mathbf{v}}| \leq 2|\text{supp}(\mathbf{v})| - 1$ holds.

To provide the bound claimed in the theorem, suppose that the histograms $H_{\mathbf{v}}$ are chosen such that their combined support $\mathbb{T} \stackrel{\text{def}}{=} \bigcup_{\mathbf{v} \in V} \text{supp}(H_{\mathbf{v}})$ is minimal. We show that this set cannot have more than $|\text{supp}(\mathbf{x})| + 1 + \sum_{\mathbf{v} \in V} (2|\text{supp}(\mathbf{v})| - 1)$ elements, which implies the claim.

Suppose towards a contradiction that $|\mathbb{T}|$ exceeds this bound. Then it must contain two distinct elements β_1 and β_2 that are both not in $\bigcup_{\mathbf{v} \in V} \mathbb{B}_{\mathbf{v}}$ nor in $\text{supp}(\mathbf{x})$. Notice that the first condition, that β_1, β_2 are not big in any histogram $H_{\mathbf{v}}$ guarantees that

$$\sum_{\alpha \in \text{supp}(\mathbf{v})} H_{\mathbf{v}}(\alpha, \beta_1) + \sum_{\alpha \in \text{supp}(\mathbf{v})} H_{\mathbf{v}}(\alpha, \beta_2) \leq n_{\mathbf{v}} \quad \text{for all } \mathbf{v} \in V. \quad (3)$$

Based on β_1 and β_2 we introduce, for each $\mathbf{v} \in V$, the function $F_{\mathbf{v}} : \text{supp}(\mathbf{v}) \times \mathbb{D} \rightarrow \mathbb{N}$ as

$$F_{\mathbf{v}}(\alpha, \beta) = \begin{cases} H_{\mathbf{v}}(\alpha, \beta_1) + H_{\mathbf{v}}(\alpha, \beta_2) & \text{if } \beta = \beta_1 \\ 0 & \text{if } \beta = \beta_2 \\ H_{\mathbf{v}}(\alpha, \beta) & \text{otherwise.} \end{cases}$$

Then for any $\alpha \in \text{supp}(\mathbf{v})$ we have $\sum_{\beta \in \mathbb{D}} F_{\mathbf{v}}(\alpha, \beta) = n_{\mathbf{v}}$ and moreover, by Equation (3), we have $\sum_{\alpha \in \text{supp}(\mathbf{v})} F_{\mathbf{v}}(\alpha, \beta_1) \leq n_{\mathbf{v}}$. So $F_{\mathbf{v}}$ is a histogram over $\text{supp}(\mathbf{v})$ of degree $n_{\mathbf{v}}$. We claim that

$$\sum_{\mathbf{v} \in V} \text{eval}(\mathbf{v}, F_{\mathbf{v}}) = \sum_{\mathbf{v} \in V} \text{eval}(\mathbf{v}, H_{\mathbf{v}}).$$

Indeed, $F_{\mathbf{v}}$ trivially satisfies $eval(\mathbf{v}, H_{\mathbf{v}}) = eval(\mathbf{v}, F_{\mathbf{v}})$ for all data except of β_1 and β_2 . Thus

$$\mathbf{x}(\beta) = \left(\sum_{\mathbf{v} \in V} \mathbf{x}_{\mathbf{v}} \right) (\beta) = \left(\sum_{\mathbf{v} \in V} eval(\mathbf{v}, F_{\mathbf{v}}) \right) (\beta) \text{ for all } \beta \notin \{\beta_1, \beta_2\}.$$

Moreover, $\beta_2 \notin \text{supp}(\mathbf{x})$ so $\mathbf{x}(\beta_2) = 0$. On the other hand $(\sum_{\mathbf{v} \in V} eval(\mathbf{v}, F_{\mathbf{v}}))(\beta_2) = 0$ as for every \mathbf{v} it holds that $eval(\mathbf{v}, F_{\mathbf{v}})(\beta_2) = 0$. Finally, $\beta_1 \notin \text{supp}(\mathbf{x})$ so $\mathbf{x}(\beta_1) = 0$. On the other hand

$$\begin{aligned} \left(\sum_{\mathbf{v} \in V} eval(\mathbf{v}, F_{\mathbf{v}}) \right) (\beta_1) &= \left(\sum_{\mathbf{v} \in V} eval(\mathbf{v}, H_{\mathbf{v}}) \right) (\beta_1) + \left(\sum_{\mathbf{v} \in V} eval(\mathbf{v}, H_{\mathbf{v}}) \right) (\beta_2) \\ &= \left(\sum_{\mathbf{v} \in V} \mathbf{x}_{\mathbf{v}} \right) (\beta_1) + \left(\sum_{\mathbf{v} \in V} \mathbf{x}_{\mathbf{v}} \right) (\beta_2) = \mathbf{x}(\beta_1) + \mathbf{x}(\beta_2) = 0 + 0 = 0. \end{aligned}$$

We conclude that $\mathbf{x} = \sum_{\mathbf{v} \in V} eval(\mathbf{v}, F_{\mathbf{v}})$. But this contradicts the minimality of $|\mathbb{T}|$ as it strictly includes $\bigcup_{\mathbf{v} \in V} \text{supp}(F_{\mathbf{v}}) = \mathbb{T} \setminus \{\beta_2\}$. ◀

► **Corollary 8.** *The Expressibility problem for data vectors with values in $(\mathbb{Z}^d, +)$ is NP-complete.*

Proof. We show the upper bound only, as a matching lower bound holds already for singleton domains \mathbb{D} , where the problem is equivalent to the feasibility of integer linear programs.

By Theorem 7, positive instances imply the existence of histograms $H_{\mathbf{v}}$ over $\text{supp}(\mathbf{v})$, with polynomially bounded support, with $\mathbf{x} = \sum_{\mathbf{v} \in V} eval(\mathbf{v}, H_{\mathbf{v}})$. By Lemma 6, the existence of such histograms is also a sufficient condition for \mathbf{x} to be a permutation sum of V .

Due to the bound from Theorem 7, the histogram conditions as well as the condition that $\mathbf{x} = \sum_{\mathbf{v} \in V} eval(\mathbf{v}, H_{\mathbf{v}})$ can be expressed as a system of linear constraints with polynomially many inequalities and unknowns, which has a non-negative integer solution iff these conditions are satisfied. The claim thus follows from standard results for integer linear programming. ◀

5 Reversibility

In this section we consider data vectors $\mathbf{v} : \mathbb{D} \rightarrow G$ where $(G, +)$ is a commutative group. In this case the set of all vectors is itself a commutative group with identity $\mathbf{0}$. We write $-\mathbf{v}$ for the inverse of vector \mathbf{v} and $\mathbf{v} - \mathbf{w} \stackrel{\text{def}}{=} \mathbf{v} + (-\mathbf{w})$.

► **Definition 9.** A vector $\mathbf{x} : \mathbb{D} \rightarrow G$ is *reversible in V* if both \mathbf{x} and $-\mathbf{x}$ are permutation sums of V . A set of vectors V is reversible if every vector $\mathbf{v} \in V$ is reversible in V .

We will provide in this section a way to reduce the Expressibility problem to the membership problems in finitely generated subgroups of $(G, +)$, assuming the given set of data vectors is reversible. This result stated as Theorem 15. We also show (as Theorem 11), that checking the reversibility condition amounts to solving a finite linear system over $(G, +)$.

Our constructions are based on the homomorphism *weight*, which projects data vectors into the underlying group: the weight of a vector $\mathbf{v} : \mathbb{D} \rightarrow G$ is the element of G defined as

$$weight(\mathbf{x}) \stackrel{\text{def}}{=} \sum_{\alpha \in \mathbb{D}} \mathbf{x}(\alpha).$$

For a set V of data vectors define $weight(V) \stackrel{\text{def}}{=} \{weight(\mathbf{v}) \mid \mathbf{v} \in V\}$.

In addition we introduce some useful notation. Fix any total order on \mathbb{D} . The *rotation* of a finite set $\mathbb{S} \subseteq \mathbb{D}$ is the permutation $rotate_{\mathbb{S}} : \mathbb{D} \rightarrow \mathbb{D}$ defined as

$$rotate_{\mathbb{S}}(\alpha) \stackrel{\text{def}}{=} \begin{cases} \min\{\mathbb{S}\}, & \text{if } \alpha = \max\{\mathbb{S}\} \\ \min\{\beta \in \mathbb{S} \mid \beta > \alpha\}, & \text{if } \max\{\mathbb{S}\} \neq \alpha \in \mathbb{S} \\ \alpha, & \text{if } \alpha \notin \mathbb{S}. \end{cases}$$

This allows to express for instance the vector $\mathbf{v} \circ rotate_{\{\alpha, \beta\}}$, which results from the vector \mathbf{v} by exchanging the values of α and β . Clearly, for any finite set $\mathbb{S} \subseteq \mathbb{D}$, the $|\mathbb{S}|$ -fold composition of $rotate_{\mathbb{S}}$ with itself is the identity on \mathbb{D} .

Finally, we introduce vectors $\mathbf{rot}_{\mathbb{S}}(\mathbf{v}) : \mathbb{D} \rightarrow \mathbb{Z}^d$ as

$$\mathbf{rot}_{\mathbb{S}}(\mathbf{v}) \stackrel{\text{def}}{=} \sum_{i=0}^{|\mathbb{S}|-1} \mathbf{v} \circ rotate_{\mathbb{S}}^i$$

where \mathbf{v} is a data vector, $supp(\mathbf{v}) \subseteq \mathbb{S} \subseteq \mathbb{D}$ is finite, and the superscripts denote i -fold iteration. It is the result of summing up all different \mathbb{S} -rotations of \mathbf{v} . This vector is useful because it is a permutation sum of \mathbf{v} that “equalizes” all values for $\alpha \in \mathbb{S}$ to $weight(\mathbf{v})$, as stated in the proposition below.

► **Proposition 10.** *Let $\mathbf{v} : \mathbb{D} \rightarrow G$ be a data vector and $\mathbb{S} \subseteq \mathbb{D}$ finite such that $supp(\mathbf{v}) \subseteq \mathbb{S}$.*

1. $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})$ is a permutation sum of $\{\mathbf{v}\}$.
2. $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})(\alpha) = weight(\mathbf{v})$ if $\alpha \in \mathbb{S}$ and $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})(\alpha) = 0$ if $\alpha \notin \mathbb{S}$.

Identifying Reversible Sets of Vectors

► **Theorem 11.** *Let V be a set of data vectors and $\mathbf{x} \in V$. Then \mathbf{x} is reversible in V if, and only if, $weight(\mathbf{x})$ is reversible in $weight(V)$, i.e., there exist $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in V$ such that $-weight(\mathbf{x}) = \sum_{i=1}^n weight(\mathbf{v}_i)$.*

Proof of Theorem 11. For the only if direction we need to show that if $-\mathbf{x} = \sum_{i=1}^n \mathbf{v}_i \circ \theta_i$ for vectors $\mathbf{v}_i \in V$ and permutations $\theta_i : \mathbb{D} \rightarrow \mathbb{D}$. Since $weight$ is a homeomorphism we observe that $-weight(\mathbf{x})$ is expressible as a sum $\sum_{i=1}^n weight(\mathbf{w}_i)$, where $\mathbf{w}_i \in V$. The claim follows from the fact that $weight(\mathbf{v}_i \circ \theta_i) = weight(\mathbf{v}_i)$ for all $\mathbf{v}_i : \mathbb{D} \rightarrow G$.

For the opposite direction assume vectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in V$ such that $-weight(\mathbf{x}) = \sum_{j=1}^n weight(\mathbf{v}_j)$ and let $\mathbb{S} \stackrel{\text{def}}{=} \bigcup_{i=1}^n supp(\mathbf{v}_i)$. First, we aim to show that

$$-\mathbf{rot}_{\mathbb{S}}(\mathbf{x}) = \sum_{j=1}^n \mathbf{rot}_{\mathbb{S}}(\mathbf{v}_j), \quad (4)$$

that is, $-\mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\alpha) = \sum_{j=1}^n \mathbf{rot}_{\mathbb{S}}(\mathbf{v}_j)(\alpha)$ for all $\alpha \in \mathbb{D}$. As $\mathbf{x} \in V$, by definition of $\mathbf{rot}_{\mathbb{S}}(\mathbf{x})$, this trivially holds for $\alpha \notin \mathbb{S}$. For the remaining $\alpha \in \mathbb{S}$, note that by point 2 of Proposition 10 we have $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})(\alpha) = weight(\mathbf{v})$ for any $\mathbf{v} \in V$. In particular this holds for \mathbf{x} and all \mathbf{v}_j . So,

$$-\mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\alpha) = -weight(\mathbf{x}) = \sum_{j=1}^n weight(\mathbf{v}_j) = \sum_{j=1}^n \mathbf{rot}_{\mathbb{S}}(\mathbf{v}_j)(\alpha) \quad (5)$$

which proves Equation (4). Unfolding the definition of $\mathbf{rot}_{\mathbb{S}}(\mathbf{x})$ we therefore see that

$$-\mathbf{rot}_{\mathbb{S}}(\mathbf{x}) = -\left(\mathbf{x} + \sum_{i=1}^{|\mathbb{S}|-1} \mathbf{x} \circ rotate_{\mathbb{S}}^i\right) = \sum_{j=1}^n \mathbf{rot}_{\mathbb{S}}(\mathbf{v}_j)$$

and consequently that $-\mathbf{x} = (\sum_{i=1}^{|\mathbb{S}|-1} \mathbf{x} \circ \text{rotate}_{\mathbb{S}}^i) + \sum_{j=1}^n \mathbf{rot}_{\mathbb{S}}(\mathbf{v}_j)$. Now, $(\sum_{i=1}^{|\mathbb{S}|-1} \mathbf{x} \circ \text{rotate}_{\mathbb{S}}^i)$ is clearly a permutation sum of $\{\mathbf{x}\}$ and thus also of V . By point 1 of Proposition 10, also $\sum_{j=1}^n \mathbf{rot}_{\mathbb{S}}(\mathbf{v}_j)$ is a permutation sum of V . We conclude that $-\mathbf{x}$ is a permutation sum of V and therefore that \mathbf{x} is reversible in V . \blacktriangleleft

► **Corollary 12.** *Let $d \in \mathbb{N}$ and V a finite set of vectors $\mathbf{v} : \mathbb{D} \rightarrow \mathbb{Z}^d$. There is a polynomial time procedure that checks if V is reversible.*

Proof. By Theorem 11, it suffices to verify for all $\mathbf{v} \in V$ that $-\text{weight}(\mathbf{v})$ is the sum of elements in $\text{weight}(V)$. In other words, we need to check for an element h of a finite set $H = \{h_1, \dots, h_k\} \subseteq \mathbb{Z}^d$, that there exist $n_1, \dots, n_k \in \mathbb{N}$ with $-h = n_1 h_1 + \dots + n_k h_k$.

We show that the condition above is satisfied if, and only if, there exists $\lambda_1, \dots, \lambda_k \in \mathbb{Q}_{\geq 0}$ such that $-h = \lambda_1 h_1 + \dots + \lambda_k h_k$. The “only if” direction is immediate. For the converse, assume factors $\lambda_1, \dots, \lambda_k \in \mathbb{Q}_{\geq 0}$ such that $-h = \lambda_1 h_1 + \dots + \lambda_k h_k$. There exists a positive integer p such that $p\lambda_j \in \mathbb{N}$ for every j and thus $-h = (p\lambda_1)h_1 + \dots + (p\lambda_k)h_k + (p-1)h$. We have proved the claim.

Now, checking if $-h = \lambda_1 h_1 + \dots + \lambda_k h_k$ has a solution in the non-negative rationals is doable in polynomial time using linear programming [3]. \blacktriangleleft

Solving Expressibility in Reversible Sets of Vectors

In the remainder of this subsection we proof Theorem 15. We start with a lemma. All constructions in this section assume $\max_{\mathbf{v} \in V} |\text{supp}(\mathbf{v})| < |\mathbb{D}|$, that there exists at least one fresh datum in the domain.

We first prove that some special data vectors are permutation sums of V . Those vectors are defined by introducing for every element $g \in G$ and every data value $\alpha \in \mathbb{D}$, the data vector $\llbracket \alpha \mapsto g \rrbracket$ defined for every $\beta \in \mathbb{D}$ by:

$$\llbracket \alpha \mapsto g \rrbracket(\beta) \stackrel{\text{def}}{=} \begin{cases} g & \text{if } \beta = \alpha \\ 0 & \text{otherwise} \end{cases}$$

► **Lemma 13.** *Let V be a finite set of data vectors such that $\bigcup_{\mathbf{v} \in V} \text{supp}(\mathbf{v}) \subsetneq \mathbb{D}$. Then, $\llbracket \beta \mapsto g \rrbracket$ is a permutation sum of V for every g in the subgroup of $(G, +)$ generated by $\text{weight}(V)$, and for every $\beta \in \mathbb{D}$.*

Proof. Since g is in the subgroup generated by $\text{weight}(V)$, there exist a sequences $\mathbf{v}_1, \dots, \mathbf{v}_n$ of elements in V such that:

$$g = \sum_{j=1}^n \text{weight}(\mathbf{v}_j)$$

Consider now the vector $\mathbf{x} \stackrel{\text{def}}{=} \sum_{j=1}^n \mathbf{v}_j$. It has three relevant properties:

1. it is a permutation sum of V ,
2. its support is contained in $\bigcup_{\mathbf{v} \in V} \text{supp}(\mathbf{v})$, and
3. it satisfies $g = \text{weight}(\mathbf{x})$.

By point 2 and the assumption that the combined support $\bigcup_{\mathbf{v} \in V} \text{supp}(\mathbf{v})$ is strictly included in \mathbb{D} , we can pick some $\alpha \notin \text{supp}(\mathbf{x})$. Let $\mathbb{S}, \mathbb{T} \subseteq \mathbb{D}$ be defined as

$$\mathbb{S} \stackrel{\text{def}}{=} \text{supp}(\mathbf{x}) \quad \text{and} \quad \mathbb{T} \stackrel{\text{def}}{=} \text{supp}(\mathbf{x}) \cup \{\alpha\}$$

Then by Proposition 10 point 1, both $\mathbf{rot}_{\mathbb{S}}(\mathbf{x})$ and $\mathbf{rot}_{\mathbb{T}}(\mathbf{x})$ are permutation sums of V . Since V is reversible, so is the inverse $-\mathbf{rot}_{\mathbb{S}}(\mathbf{x})$. It remains to observe that

$$\llbracket \alpha \mapsto g \rrbracket = \mathbf{rot}_{\mathbb{T}}(\mathbf{x}) - \mathbf{rot}_{\mathbb{S}}(\mathbf{x}).$$

Indeed, for all $\delta \notin \mathbb{T}$ we have $\llbracket \alpha \mapsto g \rrbracket(\delta) = \mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\delta) = \mathbf{rot}_{\mathbb{T}}(\mathbf{x})(\delta) = 0$. For all $\delta \in \mathbb{S}$, by Proposition 10 point 2, it holds that $\mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\delta) = \mathbf{rot}_{\mathbb{T}}(\mathbf{x})(\delta) = \text{weight}(\mathbf{x}) = g$ and therefore that $\llbracket \alpha \mapsto g \rrbracket(\delta) = \mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\delta) - \mathbf{rot}_{\mathbb{T}}(\mathbf{x})(\delta) = g - g = 0$. For the last case that $\delta = \alpha \in \mathbb{T} \setminus \mathbb{S}$, again by Proposition 10 point 2, we have $\llbracket \alpha \mapsto g \rrbracket(\delta) = \mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\delta) - \mathbf{rot}_{\mathbb{T}}(\mathbf{x})(\delta) = g - 0 = g$. Now, the vector $\llbracket \beta \mapsto g \rrbracket = \llbracket \alpha \mapsto g \rrbracket \circ \text{rotate}_{\{\alpha, \beta\}}$ which completes the proof \blacktriangleleft

► **Lemma 14.** *Let V be a finite, reversible set of data vectors such that $\bigcup_{\mathbf{v} \in V} \text{supp}(\mathbf{v}) \subsetneq \mathbb{D}$. The data vector $\llbracket \alpha \mapsto g \rrbracket - \llbracket \beta \mapsto g \rrbracket$ is a permutation sum of V for every g in the subgroup of $(G, +)$ generated by $\{\mathbf{v}(\delta) \mid \delta \in \mathbb{D}, \mathbf{v} \in V\}$, and for every $\alpha, \beta \in \mathbb{D}$.*

Proof. Let $\mathbb{T} \stackrel{\text{def}}{=} \bigcup_{\mathbf{v} \in V} \text{supp}(\mathbf{v})$ and assume w.l.o.g. that $\alpha \neq \beta$ since otherwise the claim is trivial. It suffices to show the claim for $\alpha \in \mathbb{T}$ and $\beta \notin \mathbb{T}$.

We first show that there exists a data vector \mathbf{x} that is a permutation sum of V such that $\mathbf{x}(\alpha) = g$ and such that $\text{supp}(\mathbf{x}) \subseteq \mathbb{T}$. Since g is in the subgroup generated by $\{\mathbf{v}(\delta) \mid \delta \in \mathbb{D}, \mathbf{v} \in V\}$, there exist vectors $\mathbf{v}_1, \dots, \mathbf{v}_n \in V$ and data values $\delta_1, \dots, \delta_n \in \mathbb{D}$ such that:

$$g = \sum_{j=1}^n \mathbf{v}_j(\delta_j).$$

We can assume without loss of generality that $\mathbf{v}_j(\delta_j)$ are not equal to zero and hence $\delta_j \in \mathbb{T}$.

Let $\theta_j \stackrel{\text{def}}{=} \text{rotate}_{\{\alpha, \delta_j\}} : \mathbb{D} \rightarrow \mathbb{D}$ be the permutation that exchanges α and δ_j and consider the vector \mathbf{x} , defined as follows.

$$\mathbf{x} = \sum_{j=1}^n \mathbf{v}_j \circ \theta_j$$

Observe that \mathbf{x} is a permutation sum of V and $\mathbf{x}(\alpha) = g$ as it was required. Moreover, since $\delta_j, \alpha \in \mathbb{T}$, we deduce that $\text{supp}(\mathbf{v}_j \circ \theta_j)$ is included in \mathbb{T} and therefore that $\text{supp}(\mathbf{x}) \subseteq \mathbb{T}$.

To show the claim, let $\theta \stackrel{\text{def}}{=} \text{rotate}_{\{\alpha, \beta\}}$ be the permutation that swaps α and β and consider the vector

$$\mathbf{y} \stackrel{\text{def}}{=} \mathbf{x} - \mathbf{x} \circ \theta.$$

For all $\delta \in \mathbb{D} \setminus \{\alpha, \beta\}$ we get $\mathbf{y}(\delta) = \mathbf{x}(\delta) - \mathbf{x}(\theta(\delta)) = \mathbf{x}(\delta) - \mathbf{x}(\delta) = 0$. Moreover, $\mathbf{y}(\alpha) = \mathbf{x}(\alpha) - \mathbf{x}(\theta(\alpha)) = g - \mathbf{x}(\beta) = g$, similarly $\mathbf{y}(\beta) = -g$. We conclude that \mathbf{y} is a permutation sum of V and $\mathbf{y} = \llbracket \alpha \mapsto g \rrbracket - \llbracket \beta \mapsto g \rrbracket$. \blacktriangleleft

We can now prove our main theorem.

► **Theorem 15.** *Let V be a finite, reversible set of data vectors with $\bigcup_{\mathbf{v} \in V} \text{supp}(\mathbf{v}) \subsetneq \mathbb{D}$. A data vector \mathbf{x} is a permutation sum of V if, and only if, the following two conditions hold.*

- *$\text{weight}(\mathbf{x})$ is in the subgroup of $(G, +)$ generated by $\{\text{weight}(\mathbf{v}) \mid \mathbf{v} \in V\}$, and*
- *$\mathbf{x}(\alpha)$ is in the subgroup of $(G, +)$ generated by $\{\mathbf{v}(\delta) \mid \delta \in \mathbb{D}, \mathbf{v} \in V\}$ for every $\alpha \in \mathbb{D}$.*

Proof. If \mathbf{x} is a permutation sum of V there exists a sequence $\mathbf{v}_1, \dots, \mathbf{v}_n$ of data vectors in V and a sequence $\theta_1, \dots, \theta_n$ of data permutations such that $\mathbf{x} = \sum_{j=1}^n \mathbf{v}_j \circ \theta_j$. We derive that $\text{weight}(\mathbf{x}) = \sum_{j=1}^n \text{weight}(\mathbf{v}_j \circ \theta_j)$. Since $\text{weight}(\mathbf{v}_j \circ \theta_j) = \text{weight}(\mathbf{v}_j)$, it follows that $\text{weight}(\mathbf{x})$ is in the group generated by $\text{weight}(V)$. Moreover, for every $\alpha \in \mathbb{D}$, we have $\mathbf{x}(\alpha) = \sum_{j=1}^n \mathbf{v}_j(\theta_j(\alpha))$. Thus $\mathbf{x}(\alpha)$ is in the group generated by $\{\mathbf{v}(\delta) \mid \delta \in \mathbb{D}, \mathbf{v} \in V\}$.

For the converse direction, assume that \mathbf{x} is a data vector satisfying the two conditions. We pick $\delta \in \mathbb{D}$. From condition 2 and Lemma 14 we derive that for every $\alpha \in \text{supp}(\mathbf{x})$, the data vector $\llbracket \alpha \mapsto \mathbf{x}(\alpha) \rrbracket - \llbracket \delta \mapsto \mathbf{x}(\alpha) \rrbracket$ is a permutation sum of V . It follows that the $\mathbf{y} \stackrel{\text{def}}{=} \sum_{\alpha \in \text{supp}(\mathbf{x})} (\llbracket \alpha \mapsto \mathbf{x}(\alpha) \rrbracket - \llbracket \delta \mapsto \mathbf{x}(\alpha) \rrbracket)$ is a permutation sum of V . Notice that this vector is equal to $\mathbf{x} - \llbracket \delta \mapsto \text{weight}(\mathbf{x}) \rrbracket$. By condition 1, Lemma 13 applies and implies that $\llbracket \delta \mapsto \text{weight}(\mathbf{x}) \rrbracket$ is a permutation sum of V . So, \mathbf{x} must be a permutation sum of V . \blacktriangleleft

► **Corollary 16.** *Let \mathbb{D} be infinite, $d \in \mathbb{N}$, and V a finite, reversible set of vectors $\mathbf{v} : \mathbb{D} \rightarrow \mathbb{Z}^d$. There is a polynomial time procedure that determines if a given target vector $\mathbf{x} : \mathbb{D} \rightarrow \mathbb{Z}^d$ is a permutation sum of V .*

► **Remark 17.** To motivate the freshness assumption, $\bigcup_{\mathbf{v} \in V} \text{supp}(\mathbf{v}) \subseteq \mathbb{D}$ consider the following example, which shows that the two conditions in the claim of Theorem 15 are not necessarily sufficient on its own.

\mathbb{D} is the finite set $\{\alpha_1, \dots, \alpha_k\}$ where $\alpha_1, \dots, \alpha_k$ are distinct and $k \geq 2$. Assume that there exists $m \in G$ such that $k \cdot m \neq 0$. We introduce $V = \{\mathbf{v}, -\mathbf{v}\}$ where $\mathbf{v} = \llbracket \alpha_1 \mapsto m \rrbracket + \dots + \llbracket \alpha_k \mapsto m \rrbracket$ and $\mathbf{x} = \llbracket \alpha_1 \mapsto (k \cdot m) \rrbracket$. Observe that \mathbf{x} satisfies the two conditions of Theorem 15. Assume by contradiction that \mathbf{x} is a permutation sum of V . Since $\mathbf{v} \circ \theta = \mathbf{v}$ for every data permutation θ it follows that $\mathbf{x} = z \cdot \mathbf{v}$ for some $z \in \mathbb{Z}$. Thus $0 = \mathbf{x}(\alpha_2) = z \cdot \mathbf{v}(\alpha_2) = z \cdot m$, and $k \cdot m = \mathbf{x}(\alpha_1) = z \cdot \mathbf{v}(\alpha_1) = z \cdot m$. Hence $k \cdot m = 0$ and we get a contradiction. Hence \mathbf{x} is not a permutation sum of V .

6 Applications

6.1 Unordered data Petri nets

Unordered data nets extend the classical model of Petri nets by allowing each token to carry a datum from a countable set \mathbb{D} . We recall the definition from [33, 32]. A multiset over some set X is a function $M : X \rightarrow \mathbb{N}$. The set X^\oplus of all multisets over X is ordered pointwise, and the multiset union of $M, M' \in X^\oplus$ is $(M \oplus M') \in X^\oplus$ with $(M \oplus M')(\alpha) \stackrel{\text{def}}{=} M(\alpha) + M'(\alpha)$ for all $\alpha \in X$. If $M \geq M'$, then the multiset difference $(M \ominus M')$ is defined as the unique $X \in X^\oplus$ with $M = M' \oplus X$.

► **Definition 18.** An unordered Petri data net (UPDN) over domain \mathbb{D} is a tuple (P, T, F) where P is a finite set of *places*, T is a finite set of *transitions* disjoint from P , and $F : (P \times T) \cup (T \times P) \rightarrow \text{Var}^\oplus$ is a *flow function* that assigns each place $p \in P$ and transition $t \in T$ a multiset of over *variables* in Var .

A *marking* is a function $M : P \rightarrow \mathbb{D}^\oplus$. Intuitively, $M(p)(\alpha)$ denotes the number of tokens of type α in place p . A transition t is *enabled* in marking M with *mode* σ if $\sigma : \text{Var} \rightarrow \mathbb{D}$ is an injection such that $\sigma(F(p, t)) \leq M(p)$ for all $p \in P$. There is a step $M \rightarrow M'$ between markings M and M' if there exists t and σ such that t is enabled in M with mode σ , and for all $p \in P$,

$$M'(p) = M(p) \ominus \sigma(F(p, t)) \oplus \sigma(F(t, p)).$$

The transitive and reflexive closure of \rightarrow is written as $\xrightarrow{*}$.

Notice that UPDN are a generalization of ordinary P/T nets, which have only one type of token, i.e. $\mathbb{D} = \{\bullet\}$.

The decidability status of the reachability problem for UPDN, which asks if $M \xrightarrow{*} M'$ holds for given markings M, M' in a given UPDN, is currently open. We will discuss here a necessary condition for positive instances, an invariant sometimes called *state equations* in the Petri net literature.

First, notice that markings in UPDN can be seen as data vectors over the monoid $(\mathbb{Z}^d, +)$. For any marking M and place p , $M(p)$ is a multiset over \mathbb{D} , i.e. a data vector $M(p) : \mathbb{D} \rightarrow \mathbb{N}$. Markings are therefore isomorphic to data vectors $M : \mathbb{D} \rightarrow \mathbb{N}^d$, where $d = |P|$.

Similarly, flow function provides multisets $F(p, t)$ and $F(t, p)$ over the variables Var , so we can associate to each transition t the corresponding data vectors $F(\bullet, t)$ and $F(t, \bullet) : \text{Var} \rightarrow \mathbb{N}^d$, defined as $F(\bullet, t)(x) \stackrel{\text{def}}{=} [F(p_1, t)(x), F(p_2, t)(x) \dots F(p_{|P|}, t)(x)]$ and analogously, $F(t, \bullet)(x) \stackrel{\text{def}}{=}$

$[F(t, p_1)(x), F(t, p_2)(x) \dots F(t, p_{|P|})(x)]$. Further, the *displacement* $\Delta(t)$ of transition t is $\Delta(t) \stackrel{\text{def}}{=} F(t, \bullet) - F(\bullet, t)$, which is a function $\Delta(t) : \text{Var} \rightarrow \mathbb{Z}^d$ over $(\mathbb{Z}^d, +)$. Now, $M \rightarrow M'$ iff there is a transition t and injection $\sigma : \text{Var} \rightarrow \mathbb{D}$ such that $M - F(\bullet, t) \circ \sigma^{-1} \geq \mathbf{0}$ and $M' = M + \Delta(t) \circ \sigma^{-1}$. A necessary condition for reachability can thus be formulated as follows.

► **Proposition 19.** *If $M \xrightarrow{*} M'$ then there exists a sequence $t_1, t_2, \dots, t_k \in T$ of transitions and a sequence $\sigma_1, \sigma_2, \dots, \sigma_k$ of injections such that $M' - M = \sum_{i=1}^k \Delta(t_i) \circ \sigma_i^{-1}$.*

We say markings M and M' satisfy the state-equation, if there are transitions and injections satisfy the condition above. A direct consequence of Corollary 8 is that one can check this condition in NP.

► **Theorem 20.** *There is an NP algorithm that checks if for any two markings of a UDPN satisfy the state equation.*

The complexity of checking state equations for UDPN thus matches that of the same problem for ordinary Petri nets (via linear programming).

For UDPN where the set of transition effects is itself reversible, Expressibility can even be decided in polynomial time.

► **Theorem 21.** *Let (P, T, F) be a UDPN such that $\{\Delta(t) \mid t \in T\}$ is reversible. Checking if two markings satisfy the state equation is in P.*

This directly follows from Corollary 16. Notice that this reversibility condition on the transition effects is a fairly natural condition. For instance, if a UDPN is reversible in the usual Petri net parlance, i.e., if its reachability relation $\xrightarrow{*}$ is symmetric, then the set $\{\Delta(t) \mid t \in T\}$ is reversible in the sense of Definition 9. Indeed, otherwise there must be some $t' \in T$ where $-\Delta(t')$ is not a permutation sum of $\{\Delta(t) \mid t \in T\}$. So there are markings M, M' and injection σ with $M' = M + \Delta(t') \circ \sigma^{-1}$ and $M' \not\xrightarrow{*} M$.

Finally, we remark that by Corollary 12, we can in polynomial time check if a given UDPN satisfies the reversibility condition.

6.2 Blind Counter Automata

Blind counter automata [15] are finite automata equipped with a number of registers that store integer values and which can be independently incremented or decremented in each step. These systems correspond to vector addition systems with states (VASS) over the integers [16], where transitions are always enabled. The model can be equipped with data in a natural way.

► **Definition 22.** An *unordered data blind counter automaton* is given by a finite labelled transition system $\mathcal{A} \stackrel{\text{def}}{=} (Q, E, L)$ where edges in E are labelled by the function L with data vectors in $\mathbb{D} \rightarrow \mathbb{Z}^k$.

A *configuration* of the automaton is a pair (q, \mathbf{v}) where $q \in Q$ is a state and $\mathbf{v} \in \mathbb{D} \rightarrow \mathbb{Z}^k$. There is a step $(q, \mathbf{f}) \xrightarrow{e} (q', \mathbf{g})$ between two configurations $(q, \mathbf{f}), (q', \mathbf{g})$ if $e = (q, q') \in E$ and $\mathbf{g} = \mathbf{f} + L((q, q')) \circ \theta$ for a permutation $\theta : \mathbb{D} \rightarrow \mathbb{D}$. The reachability relation is a transitive closure of the step relation and we denote it by $\xrightarrow{*}$. Finally, a sequence of configurations and transitions of a form $(q_0, \mathbf{x}_0) \xrightarrow{e_1} (q_1, \mathbf{x}_1) \xrightarrow{e_2} \dots \xrightarrow{e_n} (q_n, \mathbf{x}_n)$ we call a *path*.

► **Theorem 23.** *The reachability problem for an unordered data blind counters automaton can be solved in NP.*

Proof. (*Sketch*). Any path Π from some initial configuration (q_0, \mathbf{x}_0) to some final configuration (q_f, \mathbf{x}_f) can be described as a skeleton path S of length at most $|Q|^2$ and a multiset C of simple cycles connected to it (like in [5, 23]). The key properties of the skeleton path are:

- it uses transitions from the path Π ,
- it visits every state that is visited by Π ,
- it starts in (q_0, \mathbf{x}_0) and ends in (q_f, \mathbf{x}) where \mathbf{x} is some data vector,
- its length is bounded by $|Q|^2$.

The fact that multiset of edges can be decomposed into a set of cycles is equivalent to the condition that for every state the number of incoming edges is equal to the number of outgoing edges in C . Having above we first guess a skeleton path S and next we solve a system of linear inequalities which binds the usage of different simple cycles with the desired effect of them.

Precisely, the algorithm first introduces $|Q|$ additional counters to our automaton and for every edge (p, q) we change label of it to $L_{new}(p, q) \stackrel{\text{def}}{=} (\mathbf{v} + \mathbf{v}_{p,q,\gamma})$ where γ is a fresh data value and $\mathbf{v}_{p,q,\gamma}$ is a data vector such that:

$$\mathbf{v}_{p,q,\gamma}(d)(\alpha) = \begin{cases} 1 & \text{if } d = q \text{ and } \alpha = \gamma \\ -1 & \text{if } d = p \text{ and } \alpha = \gamma \\ 0 & \text{otherwise.} \end{cases}$$

Now we guess the skeleton path, and in addition an instantiations of labels taken along the edges of the skeleton path. Suppose that the skeleton path is $q_0 \xrightarrow{e_1} q_1 \xrightarrow{e_2} \dots \xrightarrow{e_n} q_f$ and the instantiations labels looks as follows $L(e_1) \circ \theta_1, L(e_2) \circ \theta_2, \dots, L(e_n) \circ \theta_n$. Let \mathbb{S} denote a set of states visited by the skeleton path. Now, what remains it to calculate the number of occurrence of edges taken in the cyclic part of our path or in other words we need to express

$$\mathbf{x} - \mathbf{x}_0 - \left(\sum_{i=1}^n L(e_i) \circ \theta_i \right)$$

as a sum $\sum_{j=1} \mathbf{v}_j \circ \theta_j$ where \mathbf{v}_j are labels of edges starting in S . This is an instance of the expressibility problem for data vectors over $(\mathbb{Z}^d, +)$, which is solvable in NP by Corollary 8. To conclude, positive instances of the reachability problem are witnessed by a skeleton path, and a solution for the resulting instance of the expressibility problem, where the base vectors are labels of edges starting in states visited by the skeleton path. ◀

References

- 1 P. A. Abdulla, K. Čerāns, B. Jonsson, and Yih-Kuen Tsay. Algorithmic analysis of programs with well quasi-ordered domains. *Inform. and Comput.*, 160(1/2):109–127, 2000.
- 2 Parosh Aziz Abdulla, Giorgio Delzanno, and Laurent Van Begin. A classification of the expressive power of well-structured transition systems. *Information and computation*, 209(3):248–279, 2011.
- 3 Bengt Aspvall and Richard E Stone. Khachiyan’s linear programming algorithm. *Journal of Algorithms*, 1(1):1 – 13, 1980.
- 4 Cem Baškocag l and Salman Kurtulan. Generalized state equation for petri nets. *WTOS*, 10(9):295–305, September 2011.
- 5 Michael Blondin, Alain Finkel, Stefan G ller, Christoph Haase, and Pierre McKenzie. Reachability in Two-Dimensional Vector Addition Systems with States Is PSPACE-Complete. In *LICS 2015*, pages 32–43, 2015.
- 6 Michael Blondin, Alain Finkel, Christoph Haase, and Serge Haddad. Approaching the coverability problem continuously. *CoRR*, abs/1510.05724, 2015.
- 7 Mikolaj Bojanczyk, Bartek Klin, and Slawomir Lasota. Automata theory in nominal sets. *Logical Methods in Computer Science*, 10(3), 2014.
- 8 Mikolaj Bojanczyk, Bartek Klin, Slawomir Lasota, and Szymon Torunczyk. Turing machines with atoms. In *LICS 2013*, pages 183–192, 2013.
- 9 Laura Bozzelli and Pierre Ganty. Complexity analysis of the backward coverability algorithm for vass. In *RP 2011*, pages 96–109, 2011.
- 10 J rg Desel and Javier Esparza. Free choice Petri nets. 40, 1995.
- 11 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173. 2012.
- 12 Catherine Dufourd, Alain Finkel, and Philippe Schnoebelen. Reset nets between decidability and undecidability. In *ICALP’98*, pages 103–115, 1998.
- 13 Sami Evangelista, Christophe Pajault, and Jean-Francois Pradat-Peyre. *FORTE 2007*, chapter A Simple Positive Flows Computation Algorithm for a Large Subclass of Colored Nets, pages 177–195. 2007.
- 14 A. Finkel and Ph. Schnoebelen. Well-structured transition systems everywhere! *Theor. Comput. Sci.*, 256(1–2):63–92, 2001.
- 15 S.A. Greibach. Remarks on blind and partially blind one-way multicounter machines. *Theoretical Computer Science*, 7(3):311 – 324, 1978.
- 16 Christoph Haase and Simon Halfon. Integer vector addition systems with states. In *RP 2014*, pages 112–124, 2014.
- 17 Serge Haddad and Claude Girault. *Advances in Petri Nets 1987*, chapter Algebraic structure of flows of a regular coloured net, pages 73–88. 1987.
- 18 Piotr Hofman, Slawomir Lasota, Ranko Lazi c, J r me Leroux, Sylvain Schmitz, and Patrick Totzke. Coverability Trees for Petri Nets with Unordered Data. In *FoSSaCS*, 2016.
- 19 Kurt Jensen. *Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use - Volume 1, Second Edition*. 1996.
- 20 Bartek Klin, Eryk Kopczynski, Joanna Ochremiak, and Szymon Toru czyk. Locally finite constraint satisfaction problems. In *LICS 2015*, pages 475–486, 2015.
- 21 S. Rao Kosaraju. Decidability of reachability in vector addition systems. In *STOC’82*, pages 267–281, 1982.
- 22 Ranko Lazi c, Tom Newcomb, Jo l Ouaknine, A.W. Roscoe, and James Worrell. Nets with tokens which carry data. *Fund. Inform.*, 88(3):251–274, 2008.
- 23 Antonia Lechner, Richard Mayr, Jo l Ouaknine, Amaury Pouly, and James Worrell. Model checking flat freeze LTL on one-counter automata. *CoRR*, abs/1606.02643, 2016.
- 24 J r me Leroux and Sylvain Schmitz. Demystifying reachability in vector addition systems. In *LICS 2015*, pages 56–67, 2015.

- 25 Richard Lipton. The reachability problem requires exponential space. Technical Report 62, Yale University, 1976.
- 26 Ernst W. Mayr. An algorithm for the general Petri net reachability problem. In *STOC'81*, pages 238–246, 1981.
- 27 T. Murata. State equation, controllability, and maximal matchings of petri nets. *IEEE Transactions on Automatic Control*, 22(3):412–416, Jun 1977.
- 28 Charles Rackoff. The covering and boundedness problems for vector addition systems. *Theor. Comput. Sci.*, 6(2):223–231, 1978.
- 29 Laura Recalde, Serge Haddad, and Manuel Silva. Continuous petri nets: Expressive power and decidability issues. In *ATVA 2007*, pages 362–377, 2007.
- 30 Wolfgang Reisig. Petri nets and algebraic specifications. *Theor. Comput. Sci.*, 80(1):1–34, 1991.
- 31 Fernando Rosa-Velardo. Ordinal recursive complexity of unordered data nets. Technical Report TR-4-14, Departamento de Sistemas Informáticos y Computación, Universidad Complutense de Madrid, 2014.
- 32 Fernando Rosa-Velardo and David de Frutos-Escrig. Decidability and complexity of Petri nets with unordered data. *Theor. Comput. Sci.*, 412(34):4439–4451, 2011.
- 33 Fernando Rosa-Velardo, María Martos-Salgado, and David de Frutos-Escrig. Accelerations for the coverability set of Petri nets with names. *Fund. Inform.*, 113(3–4):313–341, 2011.
- 34 Karsten Schmidt. *Application and Theory of Petri Nets 1997: 18th International Conference, ICATPN'97 Toulouse, France, June 23–27, 1997 Proceedings*, chapter Verification of siphons and traps for algebraic Petri nets, pages 427–446. 1997.
- 35 Sylvain Schmitz and Philippe Schnoebelen. The power of well-structured systems. In *CONCUR 2013*, pages 5–24, 2013.
- 36 Manuel Silva, Enrique Terue, and José Manuel Colom. *Lectures on Petri Nets I: Basic Models: Advances in Petri Nets*, chapter Linear algebraic and linear programming techniques for the analysis of place/transition net systems, pages 309–373. 1998.
- 37 Marvin Triebel and Jan Sürmeli. Homogeneous equations of algebraic petri nets. *CoRR*, abs/1606.05490, 2016.